Class:	CPE300L	Semester:	Fall 2021

Points		Document author:	Jerrod Batu
		Author's email:	batuj1@unlv.nevada.edu
		Document topic:	Postlab 9

Instructor's comments:

1. Introduction / Theory of Operation

In this lab, I will become familiar with the design of a complete CPU by implementing and experimenting with single cycle implementation of MIPS instructions.

a. The **main elements** of a single cycle datapath include the control unit, datapath, data memory, and instruction memory. The control unit will decode the instructions and provide control signals for the datapath. The datapath will receive the control signals from the control unit to write data and calculate address into the data memory or to provide PC (program counter) address to instruction memory. The data memory receives the address and data from the datapath and outputs the data to be decoded from the control unit. The instruction memory outputs instruction data to be read into the datapath.

b. The control unit circuit which generates the control signals is **combinational**. The control unit circuit is combinational because it combines two decoders: main decoder and alu decoder. This control unit will also produce the AND gate control signal for the program counter source. These decoders will determine which instruction to be performed given by the instruction memory. Another reason why this control unit circuit is combinational is because it is does not have a clock, reset, or start input; therefore, it is not an FSM that is dependent on the clock cycle, but it is rather a set of combinational units to determine which instruction to be performed whether it be add, sub, addi, and, or, slt, beq, or j instructions.

c. There are a few **other functional units that are to be added to the ALU.** Since the ALUop is decoded with two bits, 00 for add and 01 for sub, 10 would perform these other functional units to the ALU. These instructions include add, addu, and, nor, or, sub, subu, xor, slt, sltu, lb, lbu, lh, lhu, lw, sb, sh, or sw. Primarily, the ALU is capable of storing and loading words into the register file. In addition, the ALU can also perform gate logic operations such as and, nor, or, and xor.

2. Prelab

https://docs.google.com/document/d/1W3bFRnSusdZvWCfRua07DCCUL3C_EYok/edit?usp=sharing&ouid=102808507017671072128&rtpof=true&sd=true

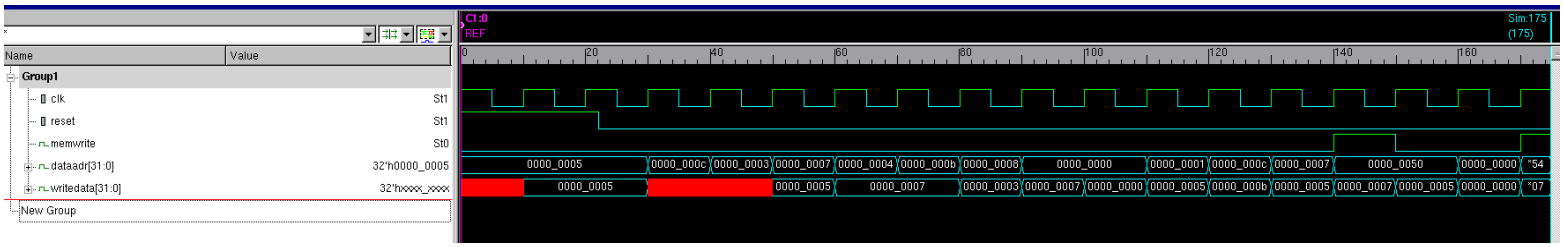
This is the link to my prelab 9 document.

3. Results of Experiments

EXPERIMENT 1

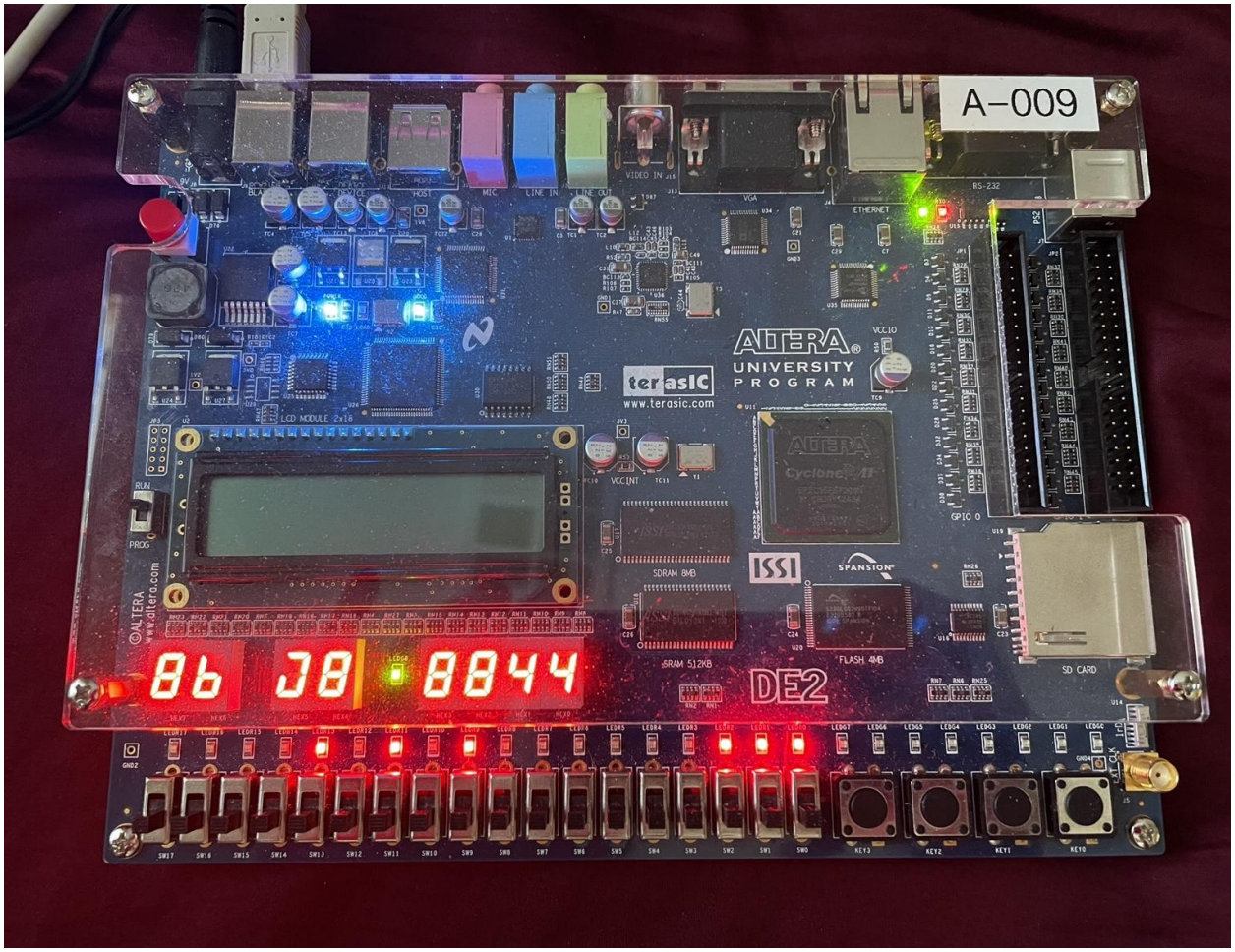
```
module testbench();
reg clk;
reg reset;
wire [31:0] writedata, dataadr;
wire memwrite;

// instantiates the  device to be tested (device under test-DUT)
MIPSsubset dut (clk, reset, writedata, dataadr, memwrite);

// initialize test
initial
begin
reset <= 1; # 22; reset <= 0;
end

// generate clock to sequence tests
always
begin
clk <= 1;
 # 5;
 clk <= 0;
 # 5; // clock duration
end

// check the contents of memory location 84. It must be 7
always @ (negedge clk)
begin
if (memwrite) begin
if (dataadr === 84 & writedata === 7) begin
$display ("Simulation succeeded");
$stop;
end
else if (dataadr !== 80) begin
$display ("Simulation failed");
$stop;
end
end
end
endmodule
```

This is my testbench module for the MIPS subset.



These are my VCS waveforms for the MIPS subset.

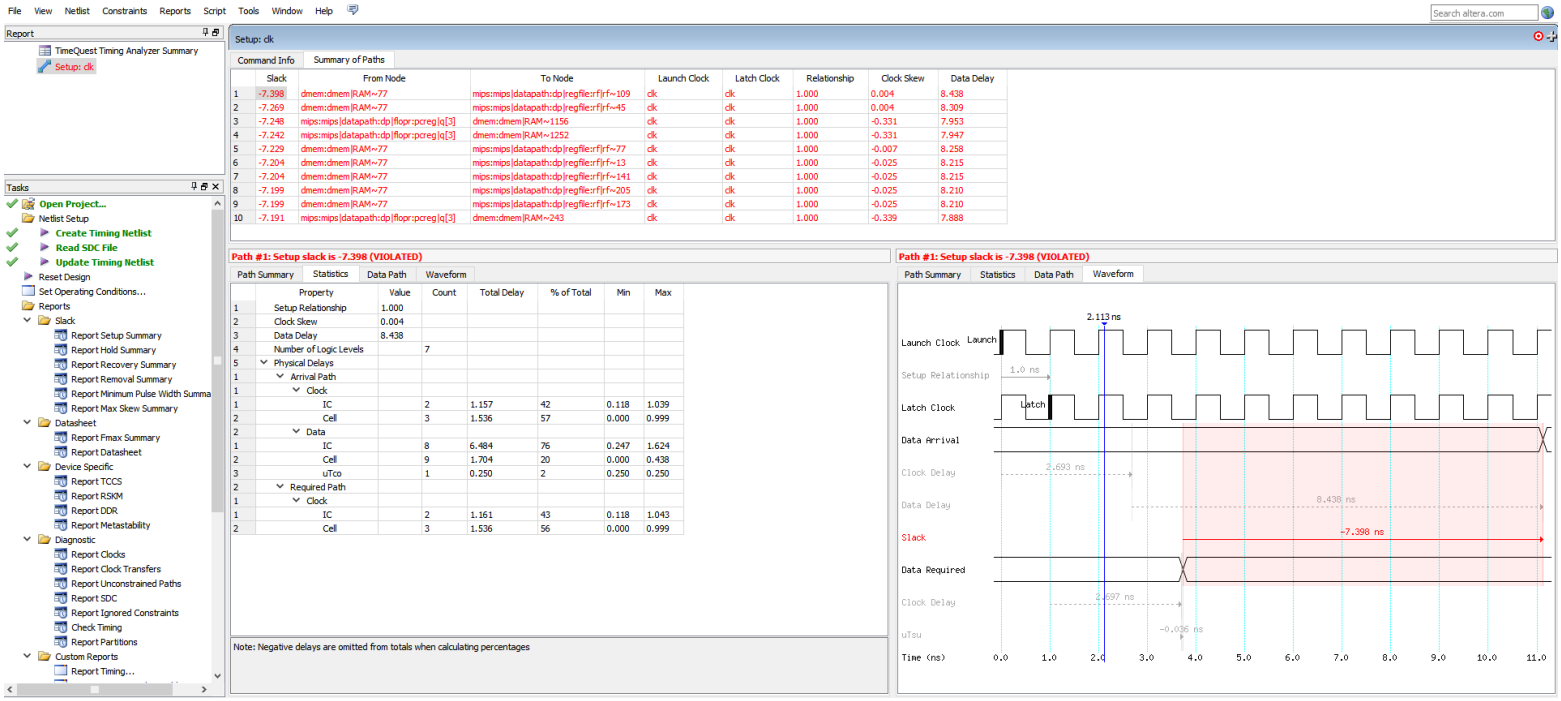
EXPERIMENT 2



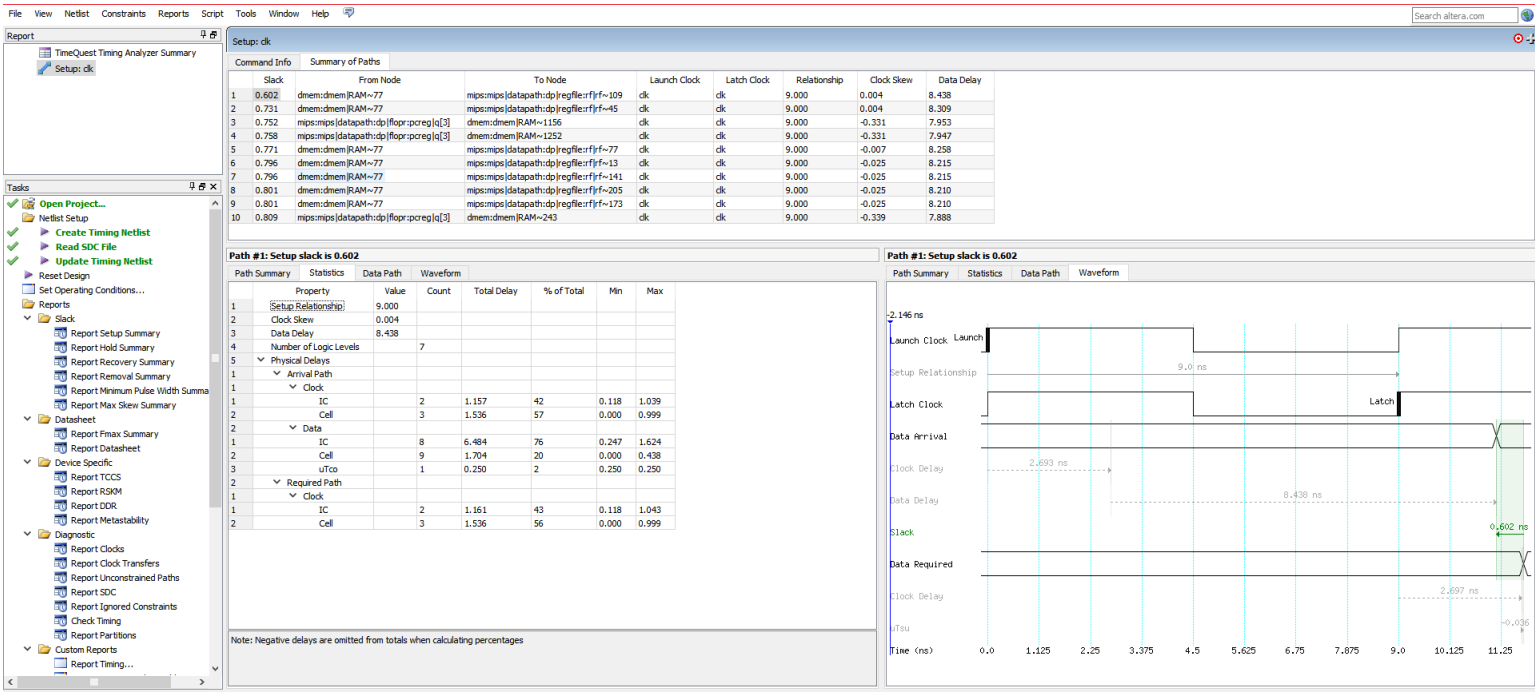
This is a picture of my DE2 board displaying the content of $2 (7) being stored into the memory location (84) at the corresponding address (44).

https://drive.google.com/file/d/1U0fXFHva_ePXsBLyd4B7FiNW13MMl7Pz/view?usp=sharing
This is the link to my video explaining each MIPS subset instruction performed on the DE2 board.

EXPERIMENT 3



This is the TimeQuest analyzer for the MIPS subset before setting the timing constraints.



This is the TimeQuest analyzer for the MIPS subset after setting the timing constraint of 9 ns.

Cycle time: 9 ns

# of instructions: 15 instructions

Total time to complete code: (9ns) x (15 instructions) = **135ns**

4. Answers to questions

Question 1:

Compared to other processor architectures, MIPS is very simple to understand and use. Since MIPS is RISC, there are a reduced instruction set of computers that are more optimized by the compilers. Compared to DEL, CISC, and VLIW, MIPS has the least amount of instruction sets. In addition, there is a flexibility of high-performance cache and memory management that allows MIPS to have a strong advantage over any other processor architectures. MIPS is also distinguished from its large number of registers, uniform instruction length, and fewer instruction formats.

Question 2:

Advantages -

1. MIPS delivers lower power consumption than other processor architectures.
2. MIPS architecture offers flexibility of high-performance caches and memory management.
3. MIPS is one of the simplest architectures to introduce computing architecture.

Disadvantages -

1. Each operation would require memory access; therefore, the processor needs to be slow or the memory must be small.
2. Instructions must encode memory addresses rather than register numbers, so the size of each instruction would need to be larger to access all memory addresses.
3. MIPS requires a mode switch before any 16-bit instructions can be executed.

5. Conclusions & Summary

Overall, this lab was very similar to what was being taught in the lecture. I am still trying to grasp the concept of the MIPS implementation in Verilog through several different components. I would say that the Verilog module instantiations are somewhat of a more complex general datapath. It is very interesting to learn how one architecture language can be implemented into a high coding language. There were not any issues or problems I had when performing the experiments for the lab, and I hope to learn more about how MIPS instructions can be incorporated into Verilog modules.